MC68000(AC1)

MOTOROLA INC.

MOS Integrated Circuits Division

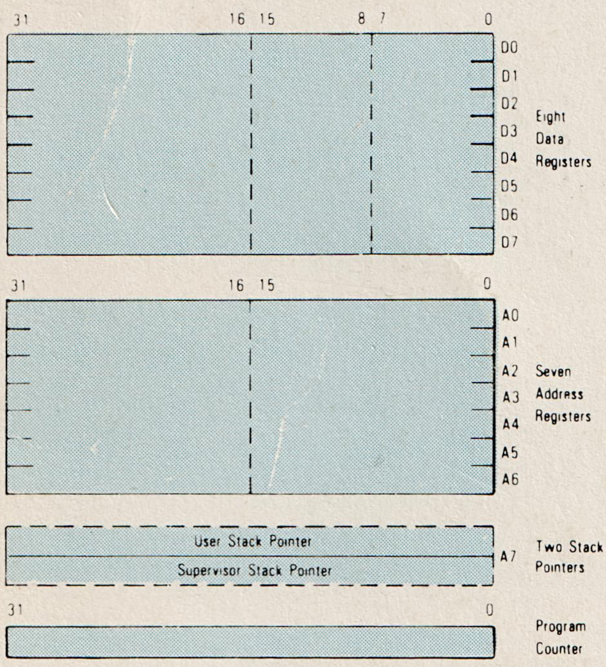
MC68000

16 Bit Microprocessor Programming Card

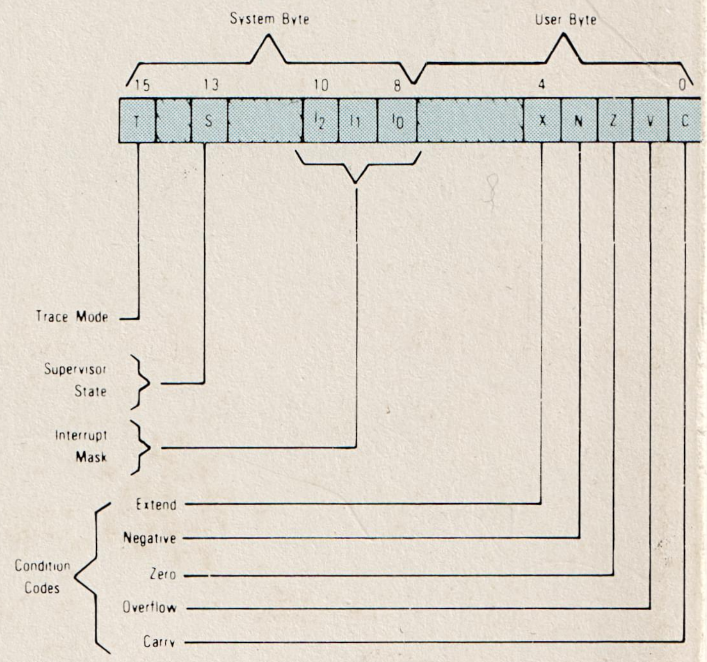
3501 Ed Bluestein Blvd. Austin, Texas 78721 Telephone: (512)928-6000

Issue A

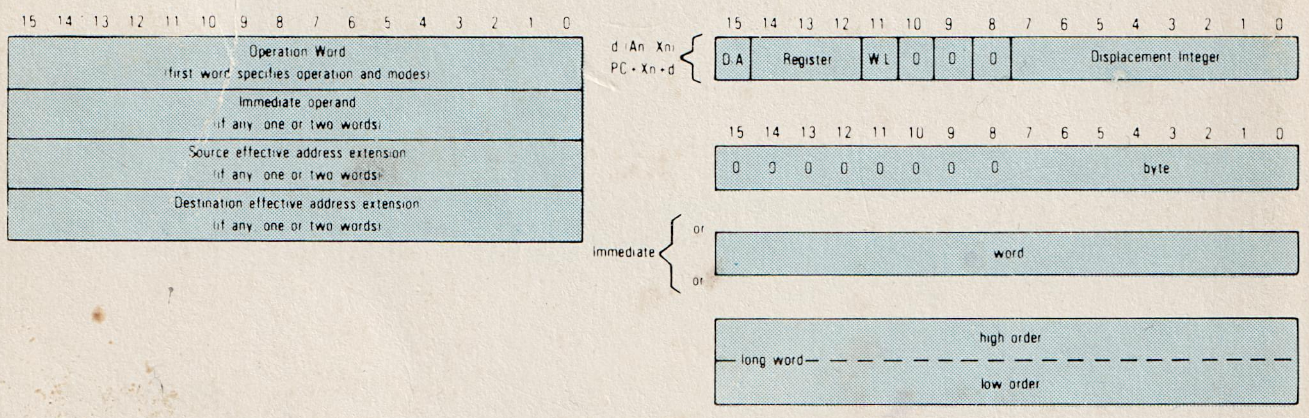
Programming Model



Status Register



Instruction Format



Single-Effective-Address-Instruction Operation Word — General Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										Effective Address					
X	X	X	X	X	X	X	X	X	X	Mode			Register		

Effective Addressing Mode Categories

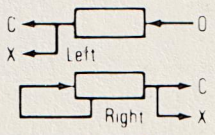
Type	Mode	Register	Generation	Assembler Syntax
Data Register Direct	000	reg. no.	EA - Dn	Dn
Address Register Direct	001	reg. no.	EA - An	An
Register Indirect	010	reg. no.	EA - (An)	(An)
Postincrement Register Indirect	011	reg. no.	EA - (An), An ← An + N	(An) +
Predecrement Register Indirect	100	reg. no.	An ← An - N, EA - (An)	- (An)
Register Indirect With Offset	101	reg. no.	EA - (An) + d16	d(An)
Indexed Register Indirect With Offset	110	reg. no.	EA - (An) + (Xn) + d8	d(An, Xn)
Absolute Short	111	000	EA - (Next Word)	xxx
Absolute Long	111	001	EA - (Next Two Words)	xxxxxx
Relative With Offset	111	010	EA - (PC) + 16	PC relative
Relative With Index and Offset	111	011	EA - (PC) + (Xn) + d8	PC relative + Xn
Immediate	111	100	Data - Next Word(s)	#xxx
Quick Immediate	–	–	Inherent Data	–
Implied Register	–	–	EA - SR, USP, SP, PC	–

Notes:
EA - Effective Address
An - Address Register
Dn - Data Register

Xn - Address or Data Register used as Index Register
SR - Status Register
PC - Program Counter
d8 - Eight bit Offset (displacement)

d16 - Sixteen bit Offset (displacement)
N - 1 for Byte, 2 for Words and 4 for Long Words
( ) - Contents of
← - Replaces

Addressing Mode

Mnemonic Operation	Size	Addr. Mode	Dn		An		(An)		(An)+		(An)		d(An)		d(An, Xi)		Abs.W		Abs.L		d(PC)		d(PC, Xi)		s=Immed d=SR/CC		Opcode Bit Pattern 1111 11		Boolean	Condition Codes
			#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	5432 1098	7654 3210		X N Z V C
ABCD	B	s-Dn d-	2	6																							1100 RRR1	0000 0rrr	d10+s10+X→d	* U * U *
Add Digits		s- -(An) d-									2	19															1100 RRR1	0000 1rrr		
ADD	B/W	s-Dn d-			ADDA		2	13	2	13	2	15	4	17	4	19	4	17	6	21							1101 DDD0	SSEE EEEE	d+Dn→d	* * * * *
Add		d-Dn s-	2	4	2*	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1101 DDD1	SSee eeee	Dn+s→Dn	
Binary	L	s-Dn d-			ADDA		2	22	2	22	2	24	4	26	4	28	4	26	6	30							1101 DDD0	10EE EEEE	d+Dn→d	
		d-Dn s-	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1101 DDD1	10ee eeee	Dn+s→Dn	
ADDA	W	d-An s-	2	8	2	8	2	12	2	12	2	14	4	16	4	18	4	16	6	20	4	16	4	18	4	12	1101 AAA0	11ee eeee	An+s→An	- - - - -
Add Address	L	d-An s-	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1101 AAA1	11ee eeee		
ADDI	B/W	s-Imm d-	4	8	ADDA		4	17	4	17	4	19	6	21	6	23	6	21	8	25							0000 0110	SSEE EEEE	d+#→d	* * * * *
Add Immed	L	s-Imm d-	6	16	ADDA		6	30	6	30	6	32	8	34	8	36	8	34	10	38										
ADDQ	B/W	s-Imm3 d-	2	4	2*	4	2	13	2	13	2	15	4	17	4	19	4	17	6	21							0101 QQQ0	SSEE EEEE	d+#→d	* * * * *
Add Quick	L	s-Imm3 d-	2	8	2	8	2	22	2	22	2	24	4	26	4	28	4	26	6	30										
ADDX	B/W	s-Dn d-	2	4																							1101 RRR1	SS00 0rrr	d+s+X→d	* * * * *
Add Multi-		s- -(An) d-									2	19															1101 RRR1	SS00 1rrr		
precision	L	s-Dn d-	2	8																							1101 RRR1	1000 0rrr		
		s- -(An) d-									2	32															1101 RRR1	1000 1rrr		
AND	B/W	s-Dn d-					2	13	2	13	2	15	4	17	4	19	4	17	6	21							1100 DDD1	SSEE EEEE	d <and> Dn→d	- * * 0 0
Logical And		d-Dn s-	2	4			2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1100 DDD0	SSee eeee	Dn <and> s→Dn	
	L	s-Dn d-					2	22	2	22	2	24	4	26	4	28	4	26	6	30							1100 DDD1	10EE EEEE	d<and>Dn→d	
		d-Dn s-	2	8			2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1100 DDD0	10ee eeee	Dn<and>s→Dn	
ANDI	B/W	s-Imm d-	4	8			4	17	4	17	4	19	6	21	6	23	6	21	8	25					4	20	0000 0010	SSEE EEEE	d<and>#→d	- * * 0 0
And Immed.	L	s-Imm d-	6	16			6	30	6	30	6	32	8	34	8	36	8	34	10	38										
ASL, ASR	B/W	count-Dn d-	2	6+2n																							1110 rrrf	SS10 0DDD		
Arithmetic		count-#1-8 d-	2	6+2n																							1110 QQQf	SS00 0DDD		* * * * *
Shift	L	count-Dn d-	2	8+2n																							1110 rrrf	1010 0DDD		
		count-#1-8 d-	2	8+2n																							1110 QQQf	1000 0DDD		
Memory	W	count-1 d-					2*	13	2*	13	2*	15	4*	17	4*	19	4*	17	6*	21							1110 000f	11EE EEEE		
BCHG	B	bit#-Dn d-					2	13	2	13	2	15	4	17	4	19	4	17	6	21							0000 rrr1	01EE EEEE	'(bit)# of d→Z,	- - * - -
Test and		bit#-Imm d-					4	17	4	17	4	19	6	21	6	23	6	21	8	25							0000 1000	01EE EEEE	'(bit)# of d→	
Change	L	bit#-Dn d-	2	<8																							0000 rrr1	01EE EEEE	(bit)# of d	
		bit#-Imm d-	4	<12																							0000 1000	01EE EEEE		
BCLR	B	bit#-Dn d-					2	13	2	13	2	15	4	17	4	19	4	17	6	21							0000 rrr1	10EE EEEE		
Test and		bit#-Imm d-					4	17	4	17	4	19	6	21	6	23	6	21	8	25							0000 1000	10EE EEEE	'(bit)# of d→Z,	- - * - -
Clear	L	bit#-Dn d-	2	<10																							0000 rrr1	10EE EEEE	0→(bit)# of d	
		bit#-Imm d-	4	<14																							0000 1000	10EE EEEE		
BSET	B	bit#-Dn d-					2	13	2	13	2	15	4	17	4	19	4	17	6	21							0000 rrr1	11EE EEEE	'(bit)# of d→Z,	- - * - -
Test and		bit#-Imm d-					4	17	4	17	4	19	6	21	6	23	6	21	8	25							0000 1000	11EE EEEE	1→(bit)# of d	
Set	L	bit#-Dn d-	2	<8																							0000 rrr1	11EE EEEE		
		bit#-Imm d-	4	<12																							0000 1000	11EE EEEE		
BTST	B	bit#-Dn d-					2	8	2	8	2	10	4	12	4	14	4	12	6	16							0000 rrr1	00EE EEEE	'(bit)# of d→Z	- - * - -
Bit Test		bit#-Imm d-					4	12	4	12	4	14	6	16	6	18	6	16	8	20							0000 1000	00EE EEEE		
	L	bit#-Dn d-	2	<6																							0000 rrr1	00EE EEEE		
		bit#-Imm d-	4	<10																							0000 1000	00EE EEEE		
CHK	W			<43	← trap →			<47		<47		<49		<51		<53		<51		<55		<51		<53		<47	0100 DDD1	10ee eeee	If Dn<0, or	- * U U U
Check Reg-		d-Dn s-	2				2		2		2		4		4		4		6		4		4		4				Dn>(bound),	
ister Against		(bound)		8'	← no →			12'		12'		14'		16'		18'		16'		20'		16'		18'		12'			then trap	
Bounds					trap																									
CLR	B/W	d-	2	4			2	13	2	13	2	15	4	17	4	19	4	17	6	21							0100 0010	SSEE EEEE	0→d	- 0 1 0 0
Clear Operand	L	d-	2	6			2	22	2	22	2	24	4	26	4	28	4	26	6	30										
CMP	B/W	d-Dn s-	2	4	2*	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1011 DDD0	SSee eeee	Dn-s	- * * * *
Compare	L	d-Dn s-	2	6	2	6	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14				
Binary																														
CMPA	W	d-An s-	2	6	2	6	2	10	2	10	2	12	4	14	4	16	4	12	6	18	4	14	4	16	4	10	1011 AAA0	11ee eeee	An-s	- * * * *
Compare	L	d-An s-	2	6	2	6	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1011 AAA1	11ee eeee		
Address																														
CMPI	B/W	s-Imm d-	4	8	CMPA		4	12	4	12	4	14	6	16	6	18	6	16	8	20							0000 1100	SSEE EEEE	d-#	- * * * *
Compare Imm.	L	s-Imm d-	6	14	CMPA		6	20	6	20	6	22	8	24	8	26	8	24	10	28										
CMPM	B/W	s-(An)+ d-							2	12																	1011 RRR1	SS00 1rrr	d-s	- * * * *
Compare	L	s-(An)+ d-							2	20																				
Memory																														
DIVS	W	d-Dn s-	2	<158			2	<162	2	<162	2	<164	4	<166	4	<168	4	<166	6	<170	4	<166	4	<168	4	<162	1000 DDD1	11ee eeee	Dn32/s16→	- * * * 0
Divide Signed																													Dn(r:q)	
DIVU	W	d-Dn s-	2	<140			2	<144	2	<144	2	<146	4	<148	4	<150	4	<148	6	<152	4	<148	4	<150	4	<144	1000 DDD0	11ee eeee	Dn32/s16→	- * * * 0
Divide																													Dn(r:q)	
Unsigned																														
EOR	B/W	s-Dn d-	2	4			2	13	2	13	2	15	4	17	4	19	4	17	6	21							1011 rrr1	SSEE EEEE	d⊕Dn→d	- * * 0 0
Exclusive OR	L	s-Dn d-	2	8			2	22	2	22	2	24	4	26	4	28	4	26	6	30										
Logical																														
EORI	B/W	s-Imm d-	4	8			4	17	4	17	4	19	6	21	6	23	6	21	8	25					4	20	0000 1010	SSEE EEEE	d⊕#→d	- * * 0 0
Exclusive OR	L	s-Imm d-	6	16			6	30	6	30	6	32	8	34	8	36	8	34	10	38										
Immediate																														
EXG	L	s-Dn d-	2	6	2	6																					1100 DDD1	0100 0DDD	s↔d	- - - - -
Exchange		s-An d-	2	6	2	6																					1100 AAA1	0100 1AAA		
Registers																											1100 DDD1	1000 1AAA		
EXT	W	d-	2	4																							0100 1000	1000 0DDD	bit 7→bit 8-15	
Sign Extend	L	d-	2	4																							0100 1000	1100 0DDD	bit 15→bit 16-31	- * * 0 0
LEA	L	d-An s-					2	4					4	8	4	12	4	8	6	12	4	8	4	12			0100 AAA1	11ee eeee	s→An	- - - - -
Load Effect-																														
ive Address																														
LINK		disp-Imm s-			4	18																					0100 1110	0101 0AAA	An→-(SP)	- - - - -
Link and																													SP→An	
Allocate																													SP+disp→SP	

*Word only
< Maximum value

Opcode Bit Pattern Key

A:	Address Register #	e:	Source Effective Address	M:	Destination EA Mode	r:	Source Register	V:	Vector #
C:	Test Condition	E:	Destination Effective Address	P:	Displacement	R:	Destination Register		
D:	Data Register #	f:	Direction; 0-Right, 1-Left	Q:	Quick Immediate Data	S:	Size; 00-B, 01-W, 10-L, 11-Another Operation		

NEG

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	0	0	Size		Effective Address					

MOVE to CCR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	0	0	1	1	Effective Address					

NOT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	1	0	Size		Effective Address					

MOVE to SR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	1	1	0	1	1	Effective Address					

NBCD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	0	0	Effective Address					

PEA

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	0	1	Effective Address					

SWAP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	0	1	0	0	0	Register		

MOVEM Registers to EA

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	1	Sz	Effective Address					

Sz: Long = 1, Word = 0

EXTW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	1	0	0	0	0	Register		

EXTL

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	0	0	1	1	0	0	0	Register		

TST

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	1	0	Size		Effective Address					

TAS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	0	1	0	1	1	Effective Address					

MOVEM EA to Registers

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	0	0	1	Sz	Effective Address					

Sz: Long = 1, Word = 0

TRAP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	0	0	Vector			

LINK

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	0	1	0	Register		

UNLK

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	0	1	1	Register		

MOVE to USP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	0	0	Register		

MOVE from USP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	0	1	Register		

RESET

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	0	0

NOP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	0	1

STOP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	1	0

RTE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	0	1	1

RTS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	1	0	1

TRAPV

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	1	1	0

RTR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	0	1	1	1	0	1	1	1

JSR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	1	0	Effective Address					

JMP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	1	1	0	1	1	Effective Address					

CHK

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	Register			1	1	0	Effective Address					

LEA

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	Register			1	1	1	Effective Address					

ADDQ

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Data			0	Size		Effective Address					

SUBQ

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Data			1	Size		Effective Address					

Scc

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Condition				1	1	Effective Address					

DBcc

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	Condition				1	1	0	0	1	Register		

Bcc

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	Condition				8-bit Displacement							

BSR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	0	0	1	8-bit Displacement							

MOVEQ

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	Register			0	Data							

OR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Register			Op-Mode			Effective Address					

Op Mode

.B	.W	.L	
000	001	010	Dn + EA → Dn
100	101	110	EA + Dn → EA

DIVU

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Register			0	1	1	Effective Address					

DIVS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Register			1	1	1	Effective Address					

SBCD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	0	Destination Register			1	0	0	0	0	R/M	Source Register		

R/M (register/memory): register-register = 0, memory-memory = 1

SUB

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	Register			Op-Mode			Effective Address					

Op Mode

.B	.W	.L	
000	001	010	Dn - EA → Dn
100	101	110	EA - Dn → EA
	011	111	An - EA → An

SUBX

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	0	1	Destination Register			1	Size		0	0	R/M	Source Register		

R/M (register/memory): register-register = 0, memory-memory = 1

CMP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	Register			Op-Mode			Effective Address					

Op-Mode

.B	.W	.L	
000	001	010	Dn-EA
	011	111	An-EA

CMPM

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	Register			1	Size		0	0	1	Register		

EOR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	1	1	Register			1	Size		Effective Address					

AND

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Register			Op-Mode			Effective Address					

Op-Mode

.B	.W	.L	
000	001	010	Dn•EA → Dn
100	101	110	EA•Dn → EA

MULU

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Register			0	1	1		Effective Address				

MULS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Register			1	1	1		Effective Address				

ABCD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Destination Register			1	0	0	0	0	R/M	Source Register		

R/M (register/memory): register-register = 0, memory-memory = 1

EXGD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Data Register			1	0	1	0	0	0	Data Register		

EXGA

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Address Register			1	0	1	0	0	1	Address Register		

EXGM

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	0	Data Register			1	1	0	0	0	1	Address Register		

ADD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	Register			Op-Mode			Effective Address					

Op-Mode

.B	.W	.L	
000	001	010	Dn + EA → Dn
100	101	110	EA + Dn → EA
	011	111	An + EA → An

ADDX

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	0	1	Destination Register			1	Size		0	0	R/M	Source Register		

R/M (register/memory): register-register = 0, memory-memory = 1

Data Register Shifts

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	Count/ Register			d	Size		i/r	Type		Register		

Memory Shifts

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	0	0	Type		d	1	1	Effective Address					

Shift Type Codes: AS = 00, LS = 01, ROX = 10, RO = 11

d (direction): Right = 0, Left = 1

i/r (count source): Immediate Count = 0, Register Count = 1

Condition Code Computations

Operations	X	N	Z	V	C	Special Definition
ABCD	*	U	?	U	?	C = Decimal Carry $Z = Z \bullet \overline{Rm} \bullet \ldots \bullet \overline{R0}$
ADD, ADDI, ADDQ	*	*	*	?	?	$V = Sm \bullet Dm \bullet \overline{Rm} + \overline{Sm} \bullet \overline{Dm} \bullet Rm$ $C = Sm \bullet Dm + \overline{Rm} \bullet Dm + Sm \bullet \overline{Rm}$
ADDX	*	*	?	?	?	$V = Sm \bullet Dm \bullet \overline{Rm} + \overline{Sm} \bullet \overline{Dm} \bullet Rm$ $C = Sm \bullet Dm + \overline{Rm} \bullet Dm + Sm \bullet \overline{Rm}$ $Z = Z \bullet \overline{Rm} \bullet \ldots \bullet \overline{R0}$
AND, ANDI, EOR, EORI, MOVEQ, MOVE, OR, ORI, CLR, EXT, NOT, TAS, TST	—	*	*	0	0	
CHK	—	*	U	U	U	
SUB, SUBI SUBQ	*	*	*	?	?	$V = \overline{Sm} \bullet Dm \bullet \overline{Rm} + Sm \bullet \overline{Dm} \bullet Rm$ $C = Sm \bullet \overline{Dm} + Rm \bullet \overline{Dm} + Sm \bullet Rm$
SUBX	*	*	?	?	?	$V = \overline{Sm} \bullet Dm \bullet \overline{Rm} + Sm \bullet \overline{Dm} \bullet Rm$ $C = Sm \bullet \overline{Dm} + Rm \bullet \overline{Dm} + Sm \bullet Rm$ $Z = Z \bullet \overline{Rm} \bullet \ldots \bullet \overline{R0}$
CMP, CMPI, CMPM	—	*	*	?	?	$V = \overline{Sm} \bullet Dm \bullet \overline{Rm} + Sm \bullet \overline{Dm} \bullet Rm$ $C = Sm \bullet \overline{Dm} + Rm \bullet \overline{Dm} + Sm \bullet Rm$
DIVS, DIVU	—	*	*	?	0	V = Division Overflow
MULS, MULU	—	*	*	0	0	
SBCD, NBCD	*	U	?	U	?	C = Decimal Borrow $Z = Z \bullet \overline{Rm} \bullet \ldots \bullet \overline{R0}$
NEG	*	*	*	?	?	$V = Dm \bullet Rm$, $C = Dm + Rm$
NEGX	*	*	?	?	?	$V = Dm \bullet Rm$, $C = Dm + Rm$ $Z = Z \bullet \overline{Rm} \bullet \ldots \bullet \overline{R0}$
BTST, BCHG, BSET, BCLR	—	—	?	—	—	$Z = \overline{Dn}$
ASL	*	*	*	?	?	$V = Dm \bullet (\overline{D_{m-1} + \ldots + D_{m-r}})$ $+ \overline{Dm} \bullet (D_{m-1} + \ldots + D_{m-r})$ $C = D_{m-r+1}$
ASL (r = 0)	—	*	*	0	0	
LSL, ROXL	*	*	*	0	?	$C = D_{m-r+1}$
LSR (r = 0)	—	*	*	0	0	
ROXL (r = 0)	—	*	*	0	?	C = X
ROL	—	*	*	0	?	$C = D_{m-r+1}$
ROL (r = 0)	—	*	*	0	0	
ASR, LSR, ROXR	*	*	*	0	?	$C = D_{r-1}$
ASR, LSR (r = 0)	—	*	*	0	0	
ROXR (r = 0)	—	*	*	0	?	C = X
ROR	—	*	*	0	?	$C = D_{r-1}$
ROR (r = 0)	—	*	*	0	0	

— Not affected
U Undefined
? Other — see Special Definition
*General Case:
X = C
N = Rm
$Z = \overline{Rm} \bullet \ldots \bullet \overline{R0}$
Sm — Source operand most significant bit
Dm — Destination operand most significant bit
Rm — Result bit most significant bit
n — bit number
r — shift amount

Powers Of 2, Powers Of 16

16^m m=	2^n n=	Value
0	0	1
	1	2
	2	4
	3	8
1	4	16
	5	32
	6	64
	7	128
2	8	256
	9	512
	10	1,024
	11	2,048
3	12	4,096
	13	8,192
	14	16,384
	15	32,768
4	16	65,536
	17	131,072
	18	262,144
	19	524,288
5	20	1,048,576
	21	2,097,152
	22	4,194,304
	23	8,388,608
6	24	16,777,216
	25	33,554,432
	26	67,108,864
	27	134,217,728
7	28	268,435,456
	29	536,870,912
	30	1,073,741,824
	31	2,147,483,648
8	32	4,294,967,296

ASCII Character Set (7-Bit Code)

LS Dig. \ MS Dig.	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	¦
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

Hexadecimal and Decimal Conversion

How to use:

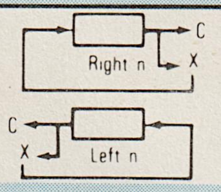
Conversion to Decimal: Find the decimal weights for corresponding hexadecimal characters beginning with the least significant character. The sum of the decimal weights is the decimal value of the hexadecimal number.

Conversion to Hexadecimal: Find the highest decimal value in the table which is lower than or equal to the decimal number to be converted. The corresponding hexadecimal character is the most significant. Subtract the decimal value found from the decimal number to be converted. With the difference repeat the process to find subsequent hexadecimal characters.

23 Byte 16				15 Byte 8				7 Byte 0			
23 Char 20		19 Char 16		15 Char 12		11 Char 8		7 Char 4		3 Char 0	
Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec	Hex	Dec
0	0	0	0	0	0	0	0	0	0	0	0
1	1,048,576	1	65,536	1	4,096	1	256	1	16	1	1
2	2,097,158	2	131,072	2	8,192	2	512	2	32	2	2
3	3,145,728	3	196,608	3	12,288	3	768	3	48	3	3
4	4,194,304	4	262,144	4	16,384	4	1024	4	64	4	4
5	5,242,880	5	327,680	5	20,480	5	1280	5	80	5	5
6	6,291,456	6	393,216	6	24,576	6	1536	6	96	6	6
7	7,340,032	7	458,752	7	28,672	7	1792	7	112	7	7
8	8,388,608	8	524,288	8	32,768	8	2048	8	128	8	8
9	9,437,184	9	589,824	9	36,864	9	2304	9	144	9	9
A	10,485,760	A	655,360	A	40,960	A	2560	A	160	A	10
B	11,534,336	B	720,896	B	45,056	B	2816	B	176	B	11
C	12,582,912	C	786,432	C	49,152	C	3072	C	192	C	12
D	13,631,488	D	851,968	D	53,248	D	3328	D	208	D	13
E	14,680,064	E	917,504	E	57,344	E	3584	E	224	E	14
F	15,728,640	F	983,040	F	61,440	F	3840	F	240	F	15

Printed in Switzerland

Addressing Mode

Mnemonic Operation	Size	Addr Mode	Dn		An		(An)		(An)+		-(An)		d(An)		d(An, Xi)		Abs.W		Abs.L		d(PC)		d(PC, Xi)		s Immed d SR/CC		Opcode Bit Pattern 1111 11	Boolean	Condition Codes
			#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	5432 1098 7654 3210		X N Z V C
ROXR, ROXL	E/W	count - Dn d-	2	6 + 2n																							1110 rrrf SS11 0DDD		• • • 0 •
Rotate		count - #1-8 d-	2	6 + 2n																							1110 QQQf SS01 0DDD		
through X	L	count - Dn d-	2	8 + 2n																							1110 rrrf 1011 0DDD		
		count - #1-8 d-	2	8 + 2n																							1110 QQQf 1001 0DDD		
Memory	W	count - 1 d-					2*	13	2*	13	2*	15	4*	17	4*	19	4*	17	6*	21							1110 010f 11EE EEEE		
SBCD	B	s - Dn d-	2	6																							1000 RRR1 0000 0rrr		
Subtract		s - -(An) d-									2	19															1000 RRR1 0000 1rrr	d10 - s10 - X → d	• U • U •
digits																													
Scc	B	cc d-	2	6/4'			2	13	2	13	2	15	4	17	4	19	4	17	6	21							0101 CCCC 11EE EEEE	If cc true, 1's → d	- - - - -
Set																												Else, 0's → d	
Conditionally																													
SUB	B/W	s - Dn d-			SUBA		2	13	2	13	2	15	4	17	4	19	4	17	6	21							1001 DDD1 SSEE EEEE	d - Dn → d	• • • • •
Subtract		d - Dn s-	2	4	2*	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1001 DDD0 SSee eeee	Dn - s → Dn	
Binary	L	s - Dn d-			SUBA		2	22	2	22	2	24	4	26	4	28	4	26	6	30							1001 DDD1 10EE EEEE	d - Dn → d	
		d - Dn s-	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1001 DDD0 10ee eeee	Dn - s → Dn	
SUBA	W	d - An s-	2	8	2	8	2	12	2	12	2	14	4	16	4	18	4	16	6	20	4	16	4	18	4	12	1001 AAA0 11ee eeee	An - s → An	- - - - -
Subtract	L	d - An s-	2	8	2	8	2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1001 AAA1 11ee eeee		
Address																													
SUBI	B/W	s - Imm d-	4	8	SUBA		4	17	4	17	4	19	6	21	6	23	6	21	8	25							0000 0100 SSEE EEEE	d - # → d	• • • • •
Subtract	L	s - Imm d-	6	16	SUBA		6	26	6	26	6	32	8	34	8	36	8	34	10	38									
Immediate																													
SUBQ	B/W	s - Imm3 d-	2	4	2*	4	2	13	2	13	2	15	4	17	4	19	4	17	6	21							0101 QQQ1 SSEE EEEE	d - # → d	• • • • •
Subtract	L	s - Imm3 d-	2	8	2	8	2	18	2	18	2	24	4	26	4	28	4	26	6	30									
Quick																													
SUBX	B/W	s - Dn d-	2	4																							1001 RRR1 SS00 0rrr	d - s - X → d	• • • • •
Subtract		s - -(An) d-									2	19															1001 RRR1 SS00 1rrr		
Multiprecision	L	s - Dn d-	2	8																							1001 RRR1 1000 0rrr		
		s - -(An) d-									2	32															1001 RRR1 1000 1rrr		
SWAP	W	d-	2	4																							0100 1000 0100 0DDD	Dn[31:16] ↔	- • • 0 0
Swap Register																												Dn[15:0]	
Halves																													
TAS	B	d-	2	4			2	15	2	15	2	17	4	19	4	21	4	19	6	23							0100 1010 11EE EEEE	test d → cc	- • • 0 0
Test and Set																												1 → bit 7 of d	
Operand																													
TST	B/W	d-	2	4			2	8	2	8	2	10	4	12	4	14	4	12	6	16							0100 1010 SSEE EEEE	test d → cc	- • • 0 0
Test	L	d-	2	4			2	12	2	12	2	14	4	16	4	18	4	16	6	20									
UNLK					2	12																					0100 1110 0101 1AAA	An → SP,	- - - - -
Unlink																												(SP)+ → An	
Program Control Instructions																													
Bcc	B	disp-																					bra taken		2	10	0110 CCCC PPPP PPPP	if cc true, PC + disp → PC	- - - - -
Branch	W	disp-																					bra not taken		2	8'			
Conditionally																							bra taken		4	10			
																							bra not taken		4	14'			
BRA	B	disp-																							2	10	0110 0000 PPPP PPPP	PC + disp → PC	- - - - -
Branch	W	disp-																							4	10			
Always																													
BSR	B	disp-																							2	20	0110 0001 PPPP PPPP	PC → -(SP),	- - - - -
Branch																												PC + disp → PC	
to Subroutine	W	disp-																							4	20			
DBcc	W	disp - Imm			cc		Counter		Branch																		0101 CCCC 1100 1DDD	If cc false,	- - - - -
Decrement				10	false		≠ -1		yes																			Dn - 1 → Dn & if	
Counter, &		counter -	4	12'	true		≠ -1		no																			Dn ≠ -1, PC + disp → PC	
Branch Until				14'	false		expired		no																			Else, NOP	
Condition																													
True or																													
Count - -1																													
JMP		d-					2	8					4	10	4	14	4	10	6	12	4	10	4	14			0100 1110 11EE EEEE	d → PC	- - - - -
Jump to																													
JSR		d-					2	18					4	20	4	24	4	20	6	22	4	20	4	24			0100 1110 10EE EEEE	PC → -(SP), d → PC	- - - - -
Jump to																													
Subroutine																													
NOP			2	4																							0100 1110 0111 0001	none	- - - - -
No Operation																													
RESET			2	132																							0100 1110 0111 0000	assert RESET pin	- - - - -
Reset External																													
Devices																													
RTE			2	20																							0100 1110 0111 0011	(SP)+ → SR,	• • • • •
Return from																												(SP)+ → PC	
Exception																													
RTR			2	20																							0100 1110 0111 0111	(SP)+ → CC,	• • • • •
Return from																												(SP)+ → PC	
Subroutine/																													
Restore CC																													
RTS			2	16																							0100 1110 0111 0101	(SP)+ → PC	- - - - -
Return from																													
Subroutine																													
STOP																									4	4	0100 1110 0111 0010	# → SR, Wait for	• • • • •
Load SR/Stop																												Interrupt	
TRAP			2	37																							0100 1110 0100 VVVV	PC → -(SSP),	- - - - -
Trap																												SR → -(SSP),	
																												(Vector) → PC	
TRAPV			2	37	Trap taken																						0100 1110 0111 0110	If V - 1, then PC →	- - - - -
Trap if				4	Trap not taken																							-(SSP), SR → -(SSP),	
Overflow Set																												(TRAPV vector) → PC,	
																												else, NOP	

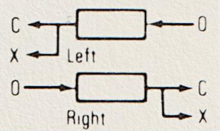
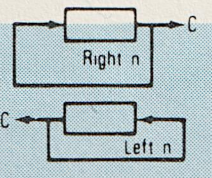
*Word only

< Maximum value

Opcode Bit Pattern Key

A:	Address Register #	e:	Source Effective Address	M:	Destination EA Mode	r:	Source Register	V:	Vector #
C:	Test Condition	E:	Destination Effective Address	P:	Displacement	R:	Destination Register		
D:	Data Register #	f:	Direction; 0 - Right, 1 - Left	Q:	Quick Immediate Data	S:	Size; 00 - B, 01 - W, 10 - L, 11 - Another Operation		

Addressing Mode

Mnemonic Operation	Size	Addr. Mode	Dn		An		(An)		(An)+		-(An)		d(An)		d(An, Xi)		Abs.W		Abs.L		d(PC)		d(PC, Xi)		s = Immed d = SR/CC		Opcode Bit Pattern 1111 11 5432 1098 7654 3210	Boolean	Condition Codes
			#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~	#	~			X N Z V C
LSL, LSR	B/W	count - Dn d-	2	6 + 2n																							1110 rrrf SS10 1DDD		· · · 0 ·
Logical Shift		count - #1-8 d-	2	6 + 2n																							1110 QQQf SS00 1DDD		
	L	count - Dn d-	2	8 + 2n																							1110 rrrf 1010 1DDD		
		count - #1-8 d-	2	8 + 2n																							1110 QQQf 1000 1DDD		
Memory	W	count - 1 d-					2*	13	2*	13	2*	15	4*	17	4*	19	4*	17	6*	21							1110 001f 11EE EEEE		
MOVE	B/W	s - Dn d-	2	4	MOVEA		2	9	2	9	2	9	4	13	4	15	4	13	6	17							00SS RRRM MMee eeee	s → d	– · · 0 0
Move Data		s - An s-	2	4	MOVEA		2	9	2	9	2	9	4	13	4	15	4	13	6	17									
		s - (An) d-	2	8	MOVEA		2	13	2	13	2	13	4	17	4	19	4	17	6	21									
		s - (An)+ d-	2	8	MOVEA		2	13	2	13	2	13	4	17	4	19	4	17	6	21									
		s - -(An) d-	2	10	MOVEA		2	15	2	15	2	15	4	19	4	21	4	19	6	23									
		s - d(An) d-	4	12	MOVEA		4	17	4	17	4	17	6	21	6	23	6	21	8	25									
		s - d(An,X) d-	4	14	MOVEA		4	19	4	19	4	19	6	23	6	25	6	23	8	27									
		s - Abs.W d-	4	12	MOVEA		4	17	4	17	4	17	6	21	6	23	6	21	8	25									
		s - Abs.L d-	6	16	MOVEA		6	21	4	21	6	21	8	25	8	27	8	25	10	29									
		s - d(PC) d-	4	12	MOVEA		4	17	4	17	4	17	6	21	6	23	6	21	8	25									
		s - d(PC,X) d-	4	14	MOVEA		4	19	4	19	4	19	6	3	6	5	6	3	8	27									
		s - Imm d-	4	8	MOVEA		4	13	4	13	4	13	6	17	6	19	6	17	8	21									
	L	s - Dn d-	2	4	MOVEA		2	14	2	14	2	~~13~~16	4	18	4	2C	4	18	6	22									
		s - An d-	2	4	MOVEA		2	14	2	14	2	~~13~~16	4	18	4	2U	4	18	6	22									
		s - (An) d-	2	12	MOVEA		2	22	2	22	2	22	4	26	4	28	4	26	6	30									
		s - (An)+ d-	2	12	MOVEA		2	22	2	22	2	22	4	26	4	28	4	26	6	30									
		s - -(An) d-	2	14	MOVEA		2	24	2	24	2	24	4	28	4	30	4	28	6	32									
		s - d(An) d-	4	16	MOVEA		4	26	4	26	4	26	6	30	6	32	6	30	8	34									
		s - d(An,X) d-	4	18	MOVEA		4	28	4	28	4	28	6	32	6	34	6	32	8	36									
		s - Abs.W d-	4	16	MOVEA		4	26	4	26	4	26	6	30	6	32	6	30	8	34									
		s - Abs.L d-	6	20	MOVEA		6	30	4	30	6	30	8	34	8	36	8	34	10	38									
		s - d(PC) d-	4	16	MOVEA		4	26	4	26	4	26	6	30	6	32	6	30	8	34									
		s - d(PC,X) d-	4	18	MOVEA		4	28	4	28	4	28	6	32	6	34	6	32	8	36									
		s - Imm d-	6	12	MOVEA		6	22	6	22	6	22	8	26	8	28	8	26	10	30									
MOVE Move to Condition Codes	W	d - CCR s-	2	12			2	16	2	16	2	18	4	20	4	22	4	20	6	24	4	20	4	22	4	16	0100 0100 11ee eeee	s → CCR	· · · · ·
MOVE	W	d - SR s-	2	12			2	16	2	16	2	18	4	20	4	22	4	20	6	24	4	20	4	22	4	16	0100 0110 11ee eeee	s → SR	· · · · ·
Move to/from Status Reg.		s - SR d-	2	6			2	13	2	13	2	15	4	17	4	19	4	17	6	21							0100 0000 11EE EEEE	SR → d	– – – – –
MOVE	L	s - USP d-			2	4																					0100 1110 0110 1AAA	USP → An	– – – – –
Move to/from User SP (A7)		d - USP s-			2	4																					0100 1110 0110 0AAA	An → USP	
MOVEA	W	d - An s-	2	4	2	4	2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	0001 AAA0 01ee eeee	s → An	– – – – –
Move Address	L	d - An s-	2	4	2	4	2	12	2	12	2	14	4	16	4	18	4	16	6	20	4	16	4	18	6	12	0010 AAA0 01ee eeee		
MOVEM Move Multiple	W	s - Xn d-					4	8 + 5n			4	8 + 5n	6	12 + 5n	6	14 + 5n	6	12 + 5n	8	16 + 5n							0100 1000 10EE EEEE a7 a0 d7 d0†	Xn → d	– – – – –
Registers		d - Xn s-					4	12 + 4n	4	12 + 4n			6	16 + 4n	6	18 + 4n	6	16 + 4n	8	20 + 4n	6	16 + 4n	6	18 + 4n			0100 1100 10ee eeee a7 a0 d7 d0	s → Xn	
	L	s - Xn d-					4	8 + 10n			4	8 + 10n	6	12 + 10n	6	14 + 10n	6	12 + 10n	8	16 + 10n							0100 1000 11EE EEEE a7 a0 d7 d0†	Xn → d	
		d - Xn s-					4	12 + 8n	4	12 + 8n			6	16 + 8n	6	18 + 8n	6	16 + 8n	8	20 + 8n	6	16 + 8n	6	18 + 8n			0100 1100 11ee eeee a7 a0 d7 d0	s → Xn	
MOVEP	W	s - Dn d-											4	18													0000 DDD1 1000 1AAA	Dn → d by bytes	– – – – –
Move		s - d(An) d-	4	16																							0000 DDD1 0000 1AAA	s → Dn by bytes	
Peripheral	L	s - Dn d-											4	28													0000 DDD1 1100 1AAA	Dn → d by bytes	
		s - d(An) d-	4	24																							0000 DDD1 0100 1AAA	s → Dn by bytes	
MOVEQ Move Quick	L	s - Imm8 d-	2	4																							0111 DDD0 QQQQ QQQQ	# → Dn	– · · 0 0
MULS Multiply Signed	W	d - Dn s-	2	< 70			2	< 74	2	< 74	2	< 76	4	< 78	4	< 80	4	< 78	6	< 82	4	< 78	4	< 80	4	< 74	1100 DDD1 11ee eeee	Dn × s → Dn	– · · 0 0
MULU Multiply Unsigned	W	d - Dn s-	2	< 70			2	< 74	2	< 74	2	< 76	4	< 78	4	< 80	4	< 78	6	< 82	4	< 78	4	< 80	4	< 74	1100 DDD0 11ee eeee	Dn × s → Dn	– · · 0 0
NBCD Negate Digit	B	d-	2	6			2	13	2	13	2	15	4	17	4	19	4	17	6	21							0100 1000 00EE EEEE	0 - d10 - X → d	· U · U ·
NEG	B/W	d-	2	4			2	13	2	13	2	15	4	17	4	19	4	17	6	21							0100 0100 SSEE EEEE	0 - d → d	· · · · ·
Negate Binary	L	d-	2	6			2	22	2	22	2	24	4	26	4	28	4	26	6	30									
NEGX	B/W	d-	2	4			2	13	2	13	2	15	4	17	4	19	4	17	6	21							0100 0000 SSEE EEEE	0 - d - X → d	· · · · ·
Negate Multi- precision	L	d-	2	6			2	22	2	22	2	24	4	26	4	28	4	26	6	30									
NOT	B/W	d-	2	4			2	13	2	13	2	15	4	17	4	19	4	17	6	21							0100 0110 SSEE EEEE	'd → d	– · · 0 0
Logical Complement	L	d-	2	6			2	22	2	22	2	24	4	26	4	28	4	26	6	30									
OR	B/W	s - Dn d-					2	13	2	13	2	15	4	17	4	19	4	17	6	21							1000 DDD1 SSEE EEEE	d < or > Dn → d	– · · 0 0
Inclusive OR		d - Dn s-	2	4			2	8	2	8	2	10	4	12	4	14	4	12	6	16	4	12	4	14	4	8	1000 DDD0 SSee eeee	Dn < or > s → Dn	
Logical	L	s - Dn d-					2	22	2	22	2	24	4	26	4	28	4	26	6	30							1000 DDD1 10EE EEEE	d < or > Dn → d	
		d - Dn s-	2	8			2	14	2	14	2	16	4	18	4	20	4	18	6	22	4	18	4	20	6	14	1000 DDD0 10ee eeee	Dn < or > s → Dn	
ORI	B/W	s - Imm d-	4	8			4	17	4	17	4	19	6	21	6	23	6	21	8	25					4	20	0000 0000 SSEE EEEE	d < or > # → d	– · · 0 0
OR Immediate	L	s - Imm d-	6	16			6	30	6	30	6	32	8	34	8	36	8	34	10	38									
PEA Push Effect- ive Address	L	s-					2	14					4	18	4	22	4	18	6	22	4	18	4	22			0100 1000 01ee eeee	s → -(SP)	– – – – –
ROR, ROL	B/W	count - Dn d-	2	6 + 2n																							1110 rrrf SS11 1DDD		– · · 0 0
Rotate		count - #1-8 d-	2	6 + 2n																									
without X	L	count - Dn d-	2	8 + 2n																							1110 QQQf SS01 1DDD		
		count - #1-8 d-	2	8 + 2n																							1110 rrrf 1011 1DDD		
	W	count - 1 d-					2*	13	2*	13	2*	15	4*	17	4*	19	4*	17	6*	21							1110 QQQf 1001 1DDD		
Memory																											1110 011f 11EE EEEE		

†Note: for PRE-DECREMENT mode, register mask d0 d7 a0 a7

*Word only

< Maximum value

Opcode Bit Pattern Key

A:	Address Register #	e:	Source Effective Address	M:	Destination EA Mode	r:	Source Register	V:	Vector #
C:	Test Condition	E:	Destination Effective Address	P:	Displacement	R:	Destination Register		
D:	Data Register #	f:	Direction; 0 - Right, 1 - Left	Q:	Quick Immediate Data	S:	Size; 00 - B, 01 - W, 10 - L, 11 - Another Operation		

Exception Vector Assignment

Vector Number(s)	Address			Assignment
	Dec	Hex	Space	
0	0	000	SP	Reset: Initial SSP
–	4	004	SP	Reset: Initial PC
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12*	48	030	SD	(Unassigned, reserved)
13*	52	034	SD	(Unassigned, reserved)
14*	56	038	SD	(Unassigned, reserved)
15	60	03C	SD	Uninitialized Interrupt Vector
16-23*	64	04C	SD	(Unassigned, reserved)
	95	05F		–
24	96	060	SD	Spurious Interrupt
25	100	064	SD	Level 1 Interrupt Autovector
26	104	068	SD	Level 2 Interrupt Autovector
27	108	06C	SD	level 3 Interrupt Autovector
28	112	070	SD	Level 4 Interrupt Autovector
29	116	074	SD	Level 5 Interrupt Autovector
30	120	078	SD	Level 6 Interrupt Autovector
31	124	07C	SD	Level 7 Interrupt Autovector
32-47	128	080	SD	TRAP Instruction Vectors
	191	0BF		–
48-63*	192	0C0	SD	(Unassigned, reserved)
	255	0FF		–
64-255	256	100	SD	User Interrupt Vectors
	1023	3FF		–

*Vector numbers 12, 13, 14, 16 through 23 and 48 through 63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.

Exception Processing Clock Periods

Exception	Periods
Address Error	57
Bus Error	57
Interrupt	47
Illegal Instruction	37
Privileged Instruction	37
Trace	37

*The interrupt acknowledge bus cycle is assumed to take four external clock periods

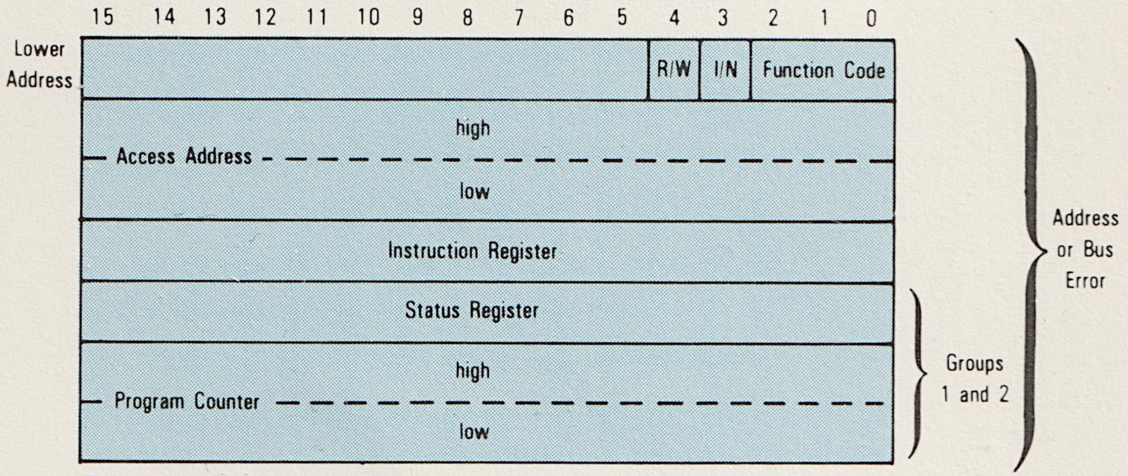
Reference Classification

Function Code Output			Reference Class
FC2	FC1	FC0	
0	0	0	(Unassigned, Reserved)
0	0	1	User Data
0	1	0	User Program
0	1	1	(Unassigned, Reserved)
1	0	0	(Unassigned, Reserved)
1	0	1	Supervisor Data
1	1	0	Supervisor Program
1	1	1	Interrupt Acknowledge

Exception Grouping and Priority

Group	Exception	Processing
0	Reset Bus Error Address Error	Exception processing begins at the next minor cycle
1	Trace Interrupt Illegal Privilege	Exception processing begins before the next instruction
2	TRAP, TRAPV, CHK Zero Divide	Exception processing is started by normal instruction execution

Supervisor Stack Order for Exception



R/W (read/write): write – 0, read – 1
I/N (instruction/not): instruction – 0, not – 1

Exception Vector Format

Word 0	New Program Counter (High)	A0 – 0, A1 – 0
Word 1	New Program Counter (Low)	A0 – 0, A1 – 1

Peripheral Vector Number Format

D15 … D8	D7							D0
Ignored	v7	v6	v5	v4	v3	v2	v1	v0

Where:
v7 is the MSB of the Vector Number
v0 is the LSB of the Vector Number

Address Translated from 8-Bit Vector Number

A23 … A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
All Zeroes	v7	v6	v5	v4	v3	v2	v1	v0	0	0

Operation Code Map

Bits 15 thru 12	Operation
0000	Bit Manipulation/MOVEP/Immediate
0001	Move Byte
0010	Move Long
0011	Move Word
0100	Miscellaneous
0101	ADDQ/SUBQ/Scc/DBcc
0110	Bcc, BSR
0111	MOVEQ
1000	OR/DIV/SBCD
1001	SUB/SUBX
1010	(Unassigned)
1011	CMP/EOR
1100	AND/MUL/ABCD/EXG
1101	ADD/ADDX
1110	Shift/Rotate
1111	(Unassigned)

Dynamic Bit

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Register			1	Type		Effective Address					

Static Bit

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	0	0	Type		Effective Address					

Bit Type Codes: TST – 00, CHG – 01, CLR – 10, SET – 11

MOVEP

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	Register			Op-Mode			0	0	1	Register		

Op Mode: Word to Reg – 100, Long to Reg – 101, Word to Mem – 110, Long to Mem – 111

OR Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	Size		Effective Address					

AND Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	1	0	Size		Effective Address					

SUB Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	0	0	Size		Effective Address					

ADD Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	Size		Effective Address					

EOR Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	0	1	0	Size		Effective Address					

CMP Immediate

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	1	0	0	Size		Effective Address					

MOVE Byte

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	Destination Register			Mode			Source Mode			Register		

MOVE Long

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	Destination Register			Mode			Source Mode			Register		

MOVE Word

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	Destination Register			Mode			Source Mode			Register		

NEGX

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0	Size		Effective Address					

MOVE from SR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	0	0	1	1	Effective Address					

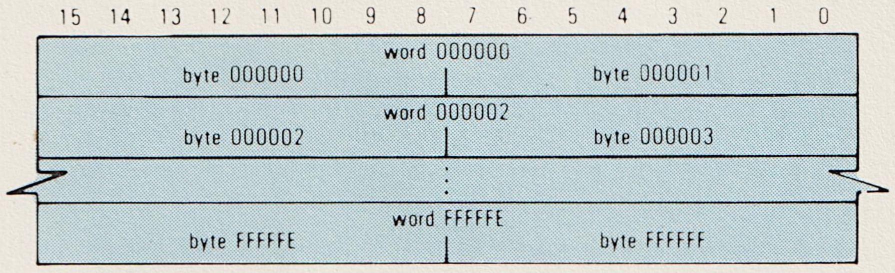
CLR

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	0	1	0	Size		Effective Address					

Conditional Tests

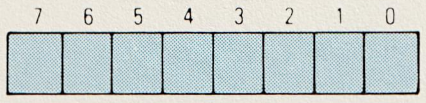
Mnemonic	Condition	Encoding	Test
T	true	0000	1
F	false	0001	0
HI	high	0010	C•Z
LS	low or same	0011	C+Z
CC	carry clear	0100	C
CS	carry set	0101	C
NE	not equal	0110	Z
EQ	equal	0111	Z
VC	overflow clear	1000	V
VS	overflow set	1001	V
PL	plus	1010	N
MI	minus	1011	N
GE	greater or equal	1100	N•V+N•V
LT	less than	1101	N•V+N•V
GT	greater than	1110	N•V•Z+N•V•Z
LE	less or equal	1111	Z+N•V+N•V

Word Organization In Memory

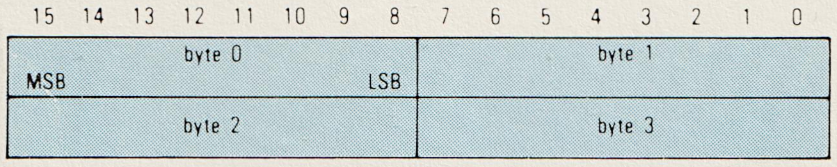


Data Organization In Memory

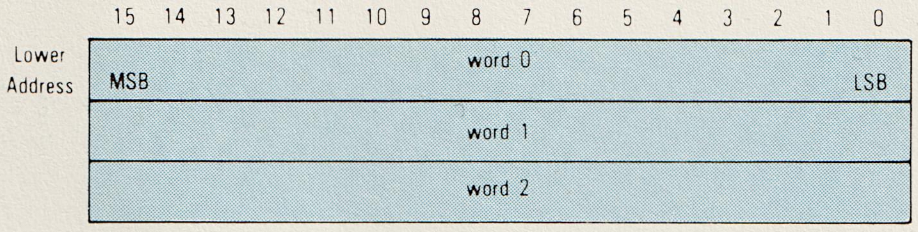
Bit Data
1 byte - 8 bits



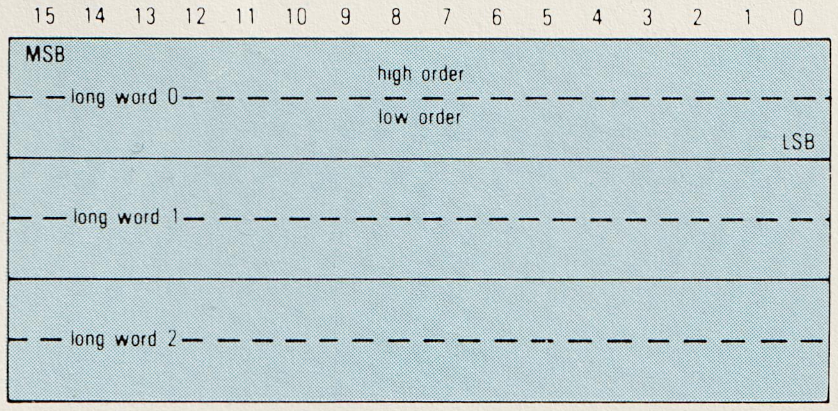
Integer Data
1 byte - 8 bits



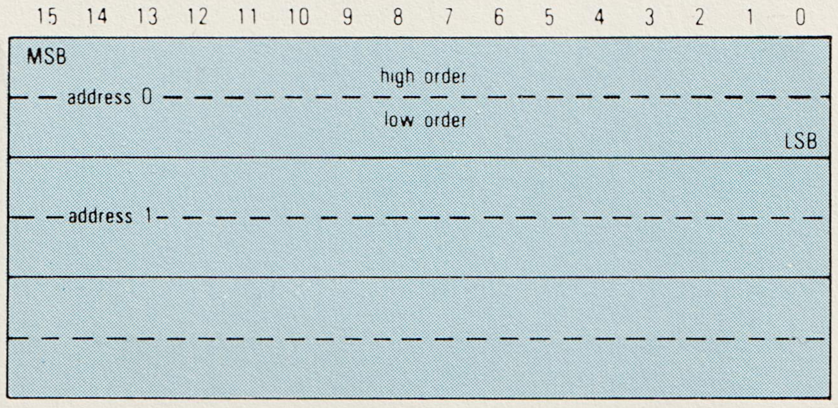
1 word - 16 bits



1 long word - 32 bits

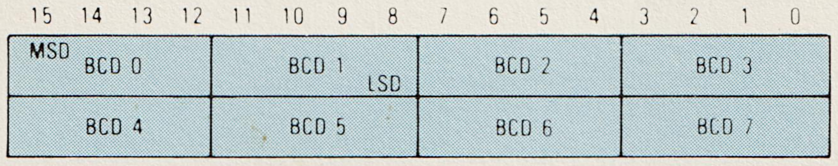


Addresses
1 address - 32 bits



MSB - Most Significant bit
LSB - Least Significant bit

Decimal Data
2 binary coded decimal digits - 1 byte



MSD - Most Significant digit
LSD - Least Significant digit

Exception Vector Assignment

Vector Number(s)	Address			Assignment
	Dec	Hex	Space	
0	0	000	SP	Reset: Initial SSP
–	4	004	SP	Reset: Initial PC
2	8	008	SD	Bus Error
3	12	00C	SD	Address Error
4	16	010	SD	Illegal Instruction
5	20	014	SD	Zero Divide
6	24	018	SD	CHK Instruction
7	28	01C	SD	TRAPV Instruction
8	32	020	SD	Privilege Violation
9	36	024	SD	Trace
10	40	028	SD	Line 1010 Emulator
11	44	02C	SD	Line 1111 Emulator
12*	48	030	SD	(Unassigned, reserved)
13*	52	034	SD	(Unassigned, reserved)
14*	56	038	SD	(Unassigned, reserved)
15	60	03C	SD	Uninitialized Interrupt Vector
16-23*	64	04C	SD	(Unassigned, reserved)
	95	05F		–
24	96	060	SD	Spurious Interrupt
25	100	064	SD	Level 1 Interrupt Autovector
26	104	068	SD	Level 2 Interrupt Autovector
27	108	06C	SD	level 3 Interrupt Autovector
28	112	070	SD	Level 4 Interrupt Autovector
29	116	074	SD	Level 5 Interrupt Autovector
30	120	078	SD	Level 6 Interrupt Autovector
31	124	07C	SD	Level 7 Interrupt Autovector
32-47	128	080	SD	TRAP Instruction Vectors
	191	0BF		–
48-63*	192	0C0	SD	(Unassigned, reserved)
	255	0FF		–
64-255	256	100	SD	User Interrupt Vectors
	1023	3FF		–

*Vector numbers 12, 13, 14, 16 through 23 and 48 through 63 are reserved for future enhancements by Motorola. No user peripheral devices should be assigned these numbers.

Exception Processing Clock Periods

Exception	Periods
Address Error	57
Bus Error	57
Interrupt	47
Illegal Instruction	37
Privileged Instruction	37
Trace	37

*The interrupt acknowledge bus cycle is assumed to take four external clock periods

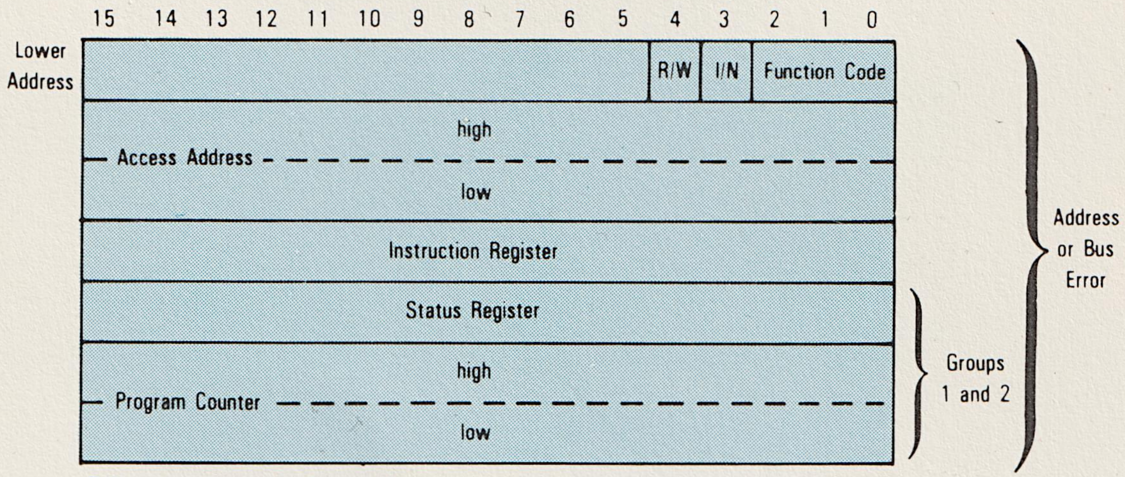
Reference Classification

Function Code Output			Reference Class
FC2	FC1	FC0	
0	0	0	(Unassigned, Reserved)
0	0	1	User Data
0	1	0	User Program
0	1	1	(Unassigned, Reserved)
1	0	0	(Unassigned, Reserved)
1	0	1	Supervisor Data
1	1	0	Supervisor Program
1	1	1	Interrupt Acknowledge

Exception Grouping and Priority

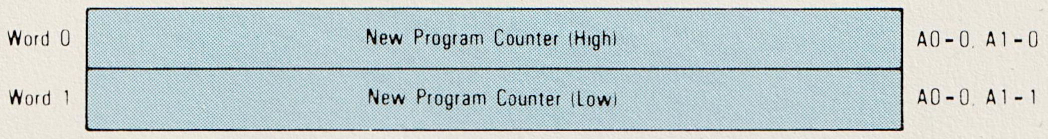
Group	Exception	Processing
0	Reset Bus Error Address Error	Exception processing begins at the next minor cycle
1	Trace Interrupt Illegal Privilege	Exception processing begins before the next instruction
2	TRAP, TRAPV, CHK Zero Divide	Exception processing is started by normal instruction execution

Supervisor Stack Order for Exception

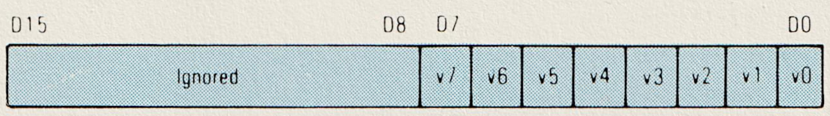


R/W (read/write): write - 0, read - 1
I/N (instruction/not): instruction - 0, not - 1

Exception Vector Format

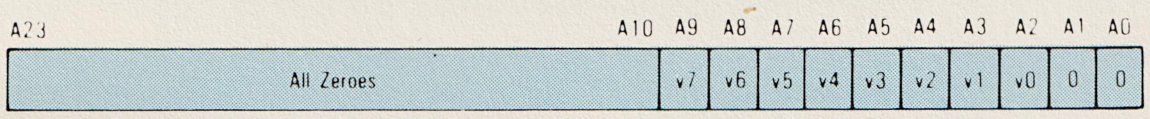


Peripheral Vector Number Format



Where:
v7 is the MSB of the Vector Number
v0 is the LSB of the Vector Number

Address Translated from 8-Bit Vector Number


A23
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
All Zeroes
v7
v6
v5
v4
v3
v2
v1
v0
0
0